Dimension-adaptive Sparse Grids

Jörg Blank

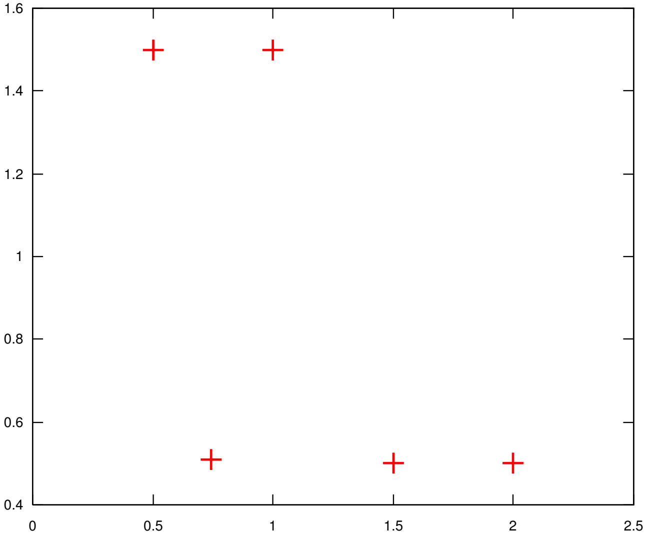
April 14, 2008

1. Data Mining
   - Function Reconstruction
2. Sparse Grids
   - Motivation
   - Introduction
3. Combination technique
   - Motivation
   - Dimension-adaptive
4. Examples
5. Outlook

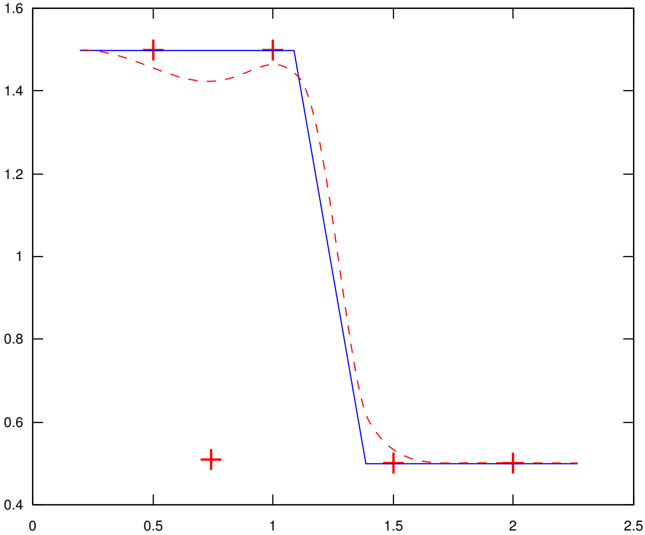
Data Mining: An use case for sparse grids

- Deduce knowledge from a (large) database
- Recover a function from test results
- Cope with measurement errors

Function Reconstruction



Function Reconstruction



Datasets

- Higher dimensions are common.

$$S = \{(x_i, y_i) \in \mathbb{R}^d \times \mathbb{R}\}_{i=1}^{M}$$

- d-dimensional dataset with M entries
- y_i - function value
- Restricting y_i to an arbitrary number of classes is possible

Datasets

- We assume that the data points are evaluations of an unknown function f

Definition

Wanted: a function

$$y = f(x_1, x_2, \ldots, x_d)$$

$$f \in V$$

where V is a function space over $\mathbb{R}^d$

Regularisation

Recover this function as good as possible!

$$\min_{f \in V} R(f)$$

$$R(f) = \frac{1}{M} \sum_{i=1}^{M} \Psi(f(\mathbf{x}_i), y_i) + \lambda \Phi(f)$$

- $\Psi(x, y) = (x - y)^2$
- $\Phi(f) = \|\nabla f\|_2^2$

Discretization

- We confine V to a discrete space V_N
- A function $f_N \in V_N$ can now be written as:

$$f_N = \sum_{j=1}^{N} \alpha_j \phi_j(\mathbf{x})$$

- weights: $\{\alpha_i\}_{i=1}^{N}$
- a base: $\Phi_N = \{\varphi_i\}_{i=1}^{N}$

The choice of basis functions has a major impact on viability and accuracy of this approach.

Differentation of $R(f_N)$ now yields for $k = 1 \ldots N$:

$$\sum_{j=1}^{N} \alpha_j \left[M\lambda(\nabla\varphi_j, \nabla\varphi_k)_{L_2} + \sum_{i=1}^{M} \varphi_j(\mathbf{x}_i) \cdot \varphi_k(\mathbf{x}_i) \right] = \sum_{i=1}^{M} y_i \varphi_k(\mathbf{x}_i)$$

Which is a system of linear equations with N unknowns and N equations and can be written in matrix form:

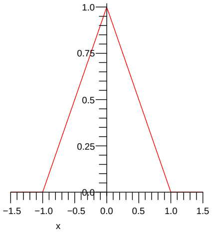
$$(\lambda C + B \cdot B^T)\alpha = By$$

This system is symmetric and positiv definite and can be solved using a standard solver like Conjugated Gradients method.

Choice of base functions

- Nodal basis yields: $O(n^d)$
- Not viable even for medium dimension counts
- Solution: Use less grid points!

Hat functions



$$\phi(x) = \begin{cases} 1 - |x| & \text{if } x \in [-1, 1] \\ 0 & \text{otherwise} \end{cases}$$

$$\phi_{l,i}(x) = \phi(\frac{x - i \cdot h_l}{h_l}) = \phi(\frac{x - i \cdot 2^{-l}}{2^{-l}}) = \phi(x \cdot 2^l - i)$$

An hierarchical basis

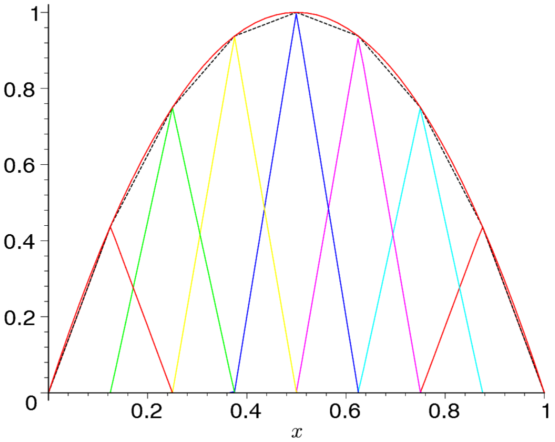


Figure: Datamining mit Dünnen Gittern, Pflüger

An hierarchical basis

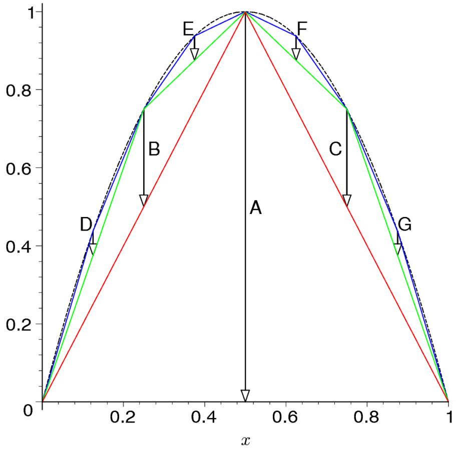


Figure: Datamining mit Dünnen Gittern, Pflüger

Pagoda

This function can be enhanced to a d-linear function

Definition

$$\phi_{\mathbf{l},\mathbf{i}}(\mathbf{x}) := \prod_{j=1}^{d} \phi_{l_j.i_j}(x_j)$$

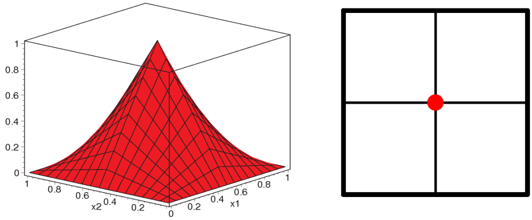


Figure: AWR2, Bungartz

Sparse Grids

Extending the hierarchical pattern yields a subgrid scheme (in 2D case):

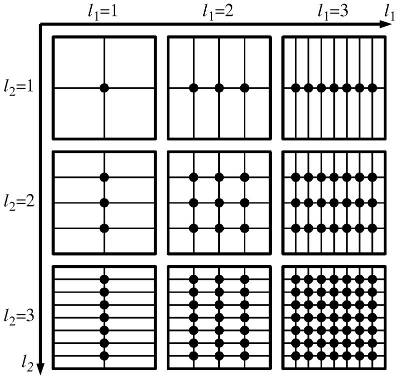
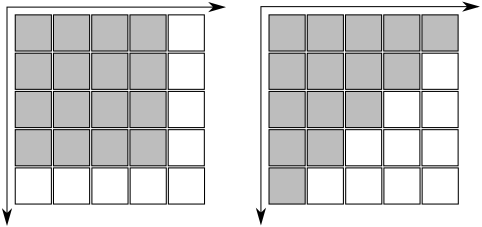


Figure: AWR2, Bungartz

Use grids with large contribution to the solution and few gridpoints



Regular Sparse Grids have a far better behaviour: $O(n * log(n)^{d-1})$

This leads to the well known pattern:

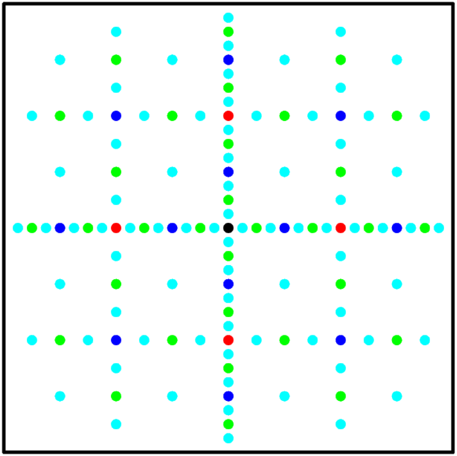


Figure: AWR2, Bungartz

Working with Sparse Grids involves a lot of overhead:

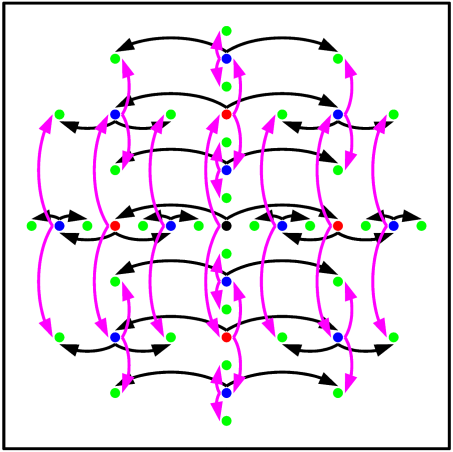


Figure: AWR, Bungartz

It is possible to create a similiar structure by combining multiple, much coarser full grids

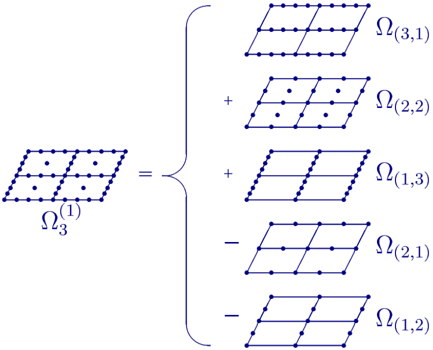
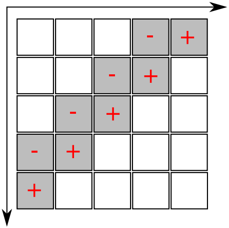


Figure: AWR2, Bungartz

For the 2D case:



$$f_n^{(c)}(\mathbf{x}) := \sum_{|\mathbf{l}|_1 = n+1} f_{\mathbf{l}}(\mathbf{x}) - \sum_{|\mathbf{l}|_1 = n} f_{\mathbf{l}}(\mathbf{x})$$

Or more general:

Definition

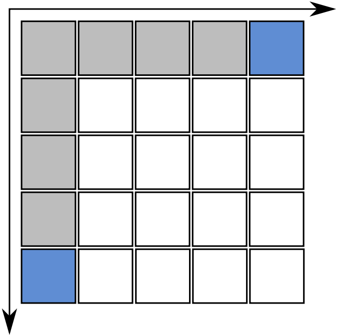
$$f_n^{(c)}(\mathbf{x}) := \sum_{q=0}^{d-1} (-1)^q \binom{d-1}{q} \sum_{|\mathbf{l}|_1 = n-q} f_{\mathbf{l}}(\mathbf{x})$$

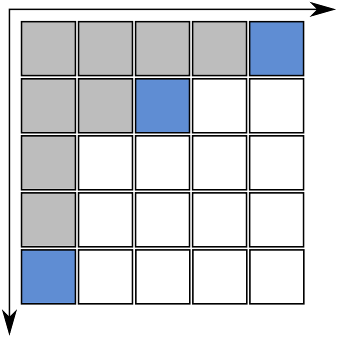
This formula is derived from the combinatorial 'inclusion-exclusion' principle!

Characteristics

- Existing codes for full grids can be used
- Embarrassingly parallel: Each subgrid can be computed without communication
- Still less points than regular nodal grids
- Only for regular sparse grids!

Generalisation





Generalisation

Allowing all subspace combinations would be a bad idea!

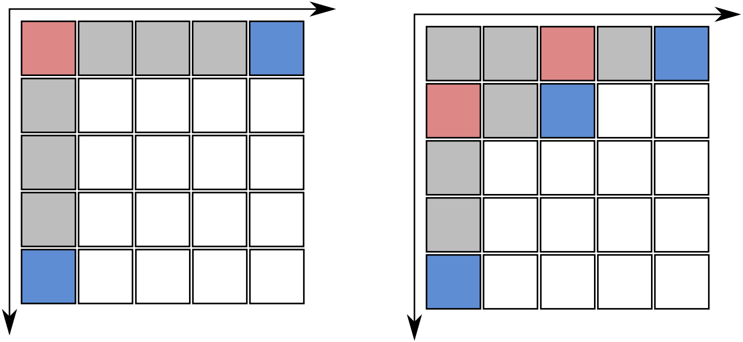
Definition

Admissibility

$\mathcal{I}$ - set of selected indices

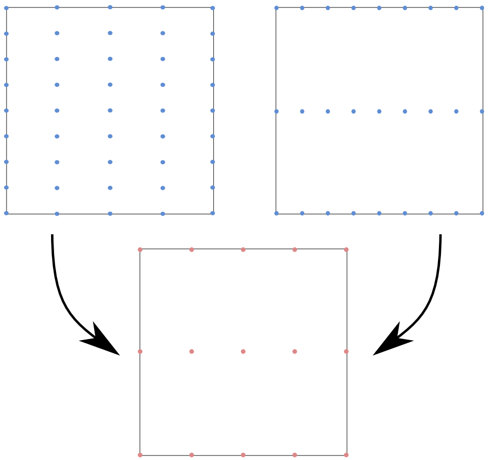
$$\underline{k} \in \mathcal{I} \text{ and } \underline{j} \leqslant \underline{k} \Rightarrow \underline{j} \in \mathcal{I}$$

Combination



Combination

Example: Combining a $(2,3)$ and a $(3,1)$ grid



Adaptivity

Start with the smallest grid: $\underline{1} = (1, \ldots, 1)$
Successively add new grid indexes:

- new index set must remain admissible
- new index must provide a significant contribution to the general solution

Error indicator

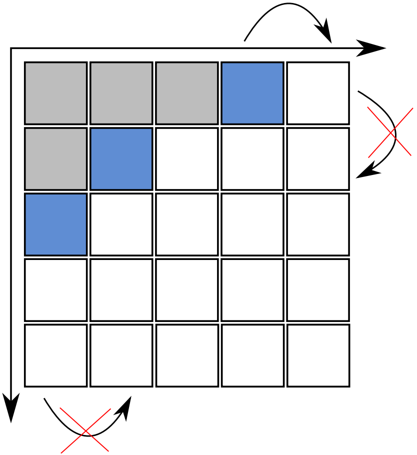
How to measure the contribution of an index?

- Calculate $\varepsilon = R(f)$
- The regularisation term may be omitted
- a large ε indicates a bad fitting $\rightarrow$ further refinement needed

Algorithm

- Initizalize index set $I = \{\underline{1}\}$
- Initizalize old index set $O = \{\}$
- Solve problem on $\underline{1}$
- while global $\varepsilon >$ bound
  - Choose $\underline{i} \in I$ with largest $\varepsilon_{\underline{i}}$
  - Refine in all dimensions, if admissible in O
  - Move $\underline{i}$ to O
  - Calculate problems and ε on new indexes
  - Update global ε

Algorithm



Why dimension-adaptivity?

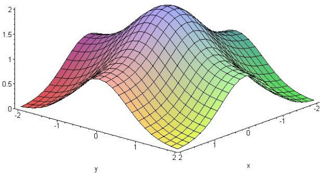
Consider:

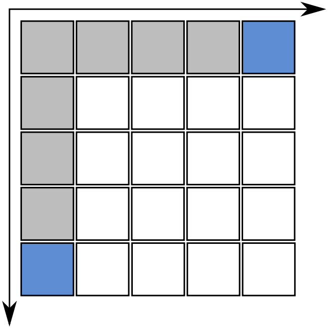
$$f(\mathbf{x}) = f_1(x_1) + f_2(x_2) + \ldots + f_d(x_d)$$

- All dimensions are totally independant
- It is possible to reconstruct the function with very little grid points
- The introduced algorithm *can* produce a near optimal result

Additive functions

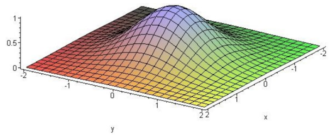
$$f(x_1, x_2) = e^{-x_1^2} + e^{-x_2^2}$$

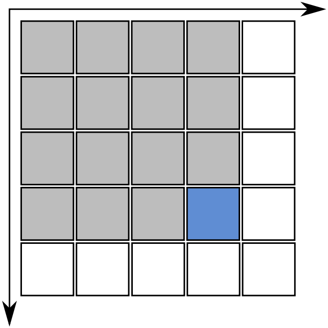




Multiplicative functions

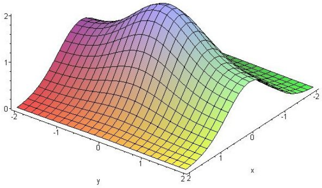
$$f(x_1, x_2) = e^{-(x_1^2+x_2^2)}$$

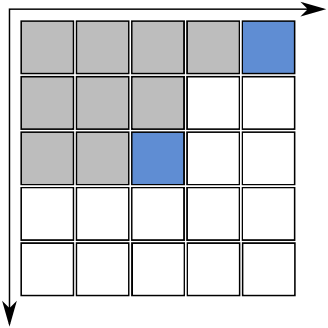




Mixed functions

$$f(x_1, x_2) = e^{-x_1^2} + e^{-(x_1^2 + x_2^2)}$$





Outlook

- Up to 15 dimensions possible in real world applications
- Or more if not to many dimensions hold informations...

- It is possible to use other coefficients for combination
  - One possibility: minimize difference to 'normal' sparse grids
  - This is called *opticom* technique by Garcke
  - Additional computional complexity, but better stability

Outlook

Other application areas:

- Integration
  An alternative for Monte-Carlo-Integration for high dimensional integrals
- Solving PDEs
  Possibility to solve PDEs in high dimensions or using a space-time-discretization

- Of course there is still the 'real' sparse grid technique

Thank you for your attention.